```
int sumto(int n)
{
  int i;
  int sum = 0;

  for (i = 0; i < n; ++i)
   sum += i;
  return sum;
}
```

Figure 1

```
int sumto(int n)
{
  int i;
  int sum = 0;

  for (i = 0; i < n; ++i)
    { sum += i; yield(); }
  return sum;
}
```

Figure 2

```
int sumto(int n)
{
  int i;
  int sum = 0;
  int tripct = 10;

  for (i = 0; i < n; ++i)
    { sum += i;
      if (0 == --tripct)
        { tripct = 10; yield(); }
    }
  return sum;
}
```

Figure 3

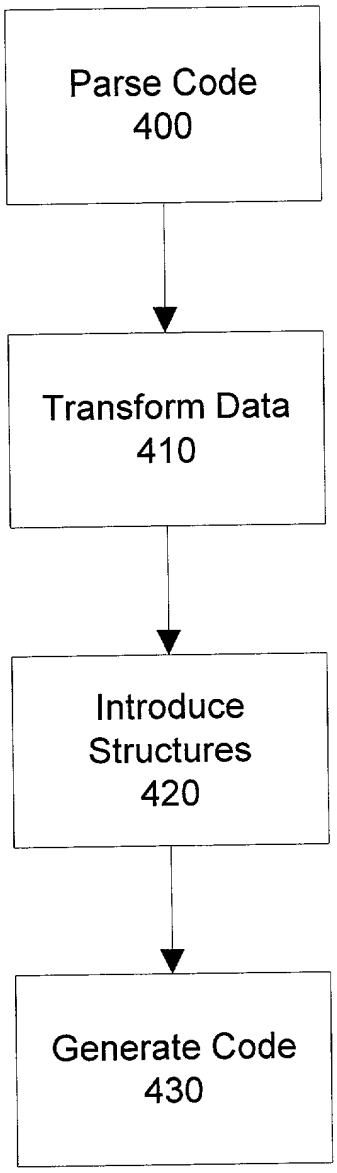
Parse Code
400
Transform Data
410
Introduce
Structures
420
Generate Code
430

Figure 4

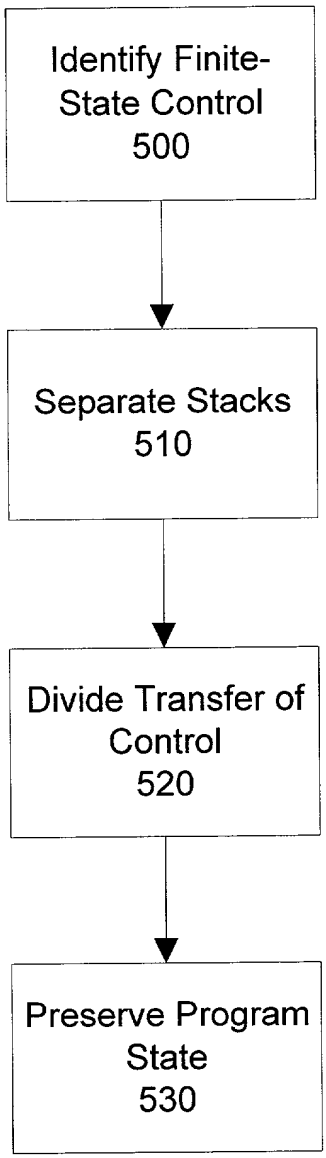
Identify Finite-State Control 500
Separate Stacks 510
Divide Transfer of Control 520
Preserve Program State 530

Figure 5

```
typedef struct context {
  int n; int i; int sum;
} context;

typedef struct Continuation {
  int state; context ctxt;
} Continuation;

bool sumto_cps(Continuation* k)
{
  int      state = k->state;
  context* ctxt  = &k->ctxt;

  switch (state) {
  case 0:  ctxt->sum = 0; ctxt->i = 0;
           k->state  = 1;
           return false;
  case 1:  { bool tst = (ctxt->i < ctxt->n);
             if (tst) k->state = 2;
             else     k->state = 3; }
           return false;
  case 2:  ctxt->sum += ctxt->i; ++ctxt->i;
           k->state = 1;
           return false;
  case 3:  return true;
  }
}

int sumto(int n) {
  Continuation k;
  bool done = false;
  k.ctxt.n = n; k.state = 0;
  while (!done)
    done = sumto_cps(&k);
  return k.ctxt.sum;
}
```

Figure 6

```
WhatNextType                                                          1
sumtoSLT (int caselab) {                                              2
  switch (caselab) {                                                  3
    ...                                                               4
    case 2:                                                           5
      arg2 = POP(void*); arg1 = POP(void*);                           6
      arg0 = POP(void*);                                              7
      kCaselab = POP(int);                                            8
      kFunc = POP(WhatNextType (*)(int ));                            9
    case 3:                                                           10
      idlab2:                                                         11
        {  int n = (int )(arg0);                                      12
           int i = (int )(arg1);                                      13
           int sum = (int )(arg2);                                    14
           if (i < n) {                                               15
              sum = sum + i; ++i;                                     16
              arg0 = (void *)(n);                                     17
              arg1 = (void *)(i);                                     18
              arg2 = (void *)(sum);                                   19
              if (--ticks > 0) goto idlab2;                           20
              PUSH(kFunc); PUSH(kCaselab);                            21
              PUSH(arg0);                                             22
              PUSH(arg1); PUSH(arg2);                                 23
              PUSH(sumtoSLT); PUSH(2);                                24
              return STOP;                                            25
           } else {                                                   26
              arg0 = (void *)(sum);                                   27
              if (--ticks > 0) goto idlab4;                           28
              PUSH(kFunc); PUSH(kCaselab);                            29
              PUSH(arg0);                                             30
              PUSH(sumtoSLT); PUSH(4);                                31
              return STOP;                                            32
           }                                                          33
        }                                                             34
        break;                                                        35
      ...                                                             36
    }                                                                 37
}                                                                     38
```

Figure 7